A Bind Control Model For Real-space Programming in Ubiquitous Computing Environment

Takuya Kashima, Shunsuke Saruwatari, Hiroyuki Morikawa, Tomonori Aoyama
The University of Tokyo
7–3–1 Hongo, Bunkyo-ku, Tokyo, 113–8656, Japan
+81 3 5841 6710
{kashima, saru, mori, aoyama}@mlab.t.u-tokyo.ac.jp

ABSTRACT

This paper describes a Bind Control Model of an ANTH (ANtennary THings) framework. ANTH is a framework to program a physical environment in ubiquitous computing environment. The Bind Control Model is one of the key technologies of ANTH framework. It is a mechanism to control bindings between objects and objects around us, and the mechanism is designed to run on tightly limited computing resource. We show the details, basic operations, and an implementation example, of the Bind Control Model.

Keywords

service synthesis, real-space programming, networked embedded system

INTRODUCTION

In ubiquitous computing environment, communication and computation functions are embedded in every object around us, and this enable us to synthesize various services by combining these objects.

Many service coordination frameworks, such as UPnP, Jini, and Bluetooth, have been proposed until now. These frameworks are useful to construct conventional services or static services, because they are designed for configuring devices automatically or replacing cable with wireless. These technologies remove annoying entwining cables and complex device configuration from us. However, they never provide us creative environment that enables us to construct various services.

In view of this, we are developing real-space programming framework called ANTH (ANtennary THings) for ubiquitous computing environment. ANTH aims to construct programmable real-space that enables us to create our desired services by ourselves. ANTH provides a chip that has three characteristic functions: a user interface function that controls connecting one device and another device, a communication function that constructs a network infrastructure for device cooperation, and a computation function which drives devices and processes applications. The chip is called ANTH chip, and an object that equips an ANTH chip is called ANTH object. We assume, in the future, ANTH chips are embedded to all everyday objects around us, such as alarm clocks, lights, walls, and so on.

The following describes a Bind Control Model that is one of the key technologies of ANTH framework.

A BIND CONTROL MODEL

ANTH object has three factors: an *emitter*, a *sympathizer*, and a *controller*. Figure 1 shows a Bind Control Model that describes the relations among the factors. Although the Bind Control Model is very simple, it enables us to make device cooperation easily and flexibly because it has the following features and advantages.

- By introduction of a *controller*, the Bind Control Model provides not only a device cooperation mechanism, but also a user interface mechanism for users to coordinate device cooperation easily.
- The communication among ANTH objects is only transmission of ANTH addresses. Therefore, ANTH does not require high bandwidth. An ANTH address is GUID (Global Unique ID). Every ANTH object has a different ANTH address as an initial value.
- In processors on ANTH nodes, every task is combined with an ANTH address. This simplicity of computing architecture can be realized in low computing resource, and gives extensibility to the ANTH node.

The following describes the details of the three factors in Figure 1, basic operations of the Bind Control Model, and the possibility of the ANTH application on this model.

ANTH Objects

EN (Emitter Node)

An EN is an event emitter like a switch, a timer of an alarm, a pressure sensor that is embedded in a chair, and so on. Each EN holds an ANTH address. When an EN detects an event, the EN emits the event packet including its ANTH address to ANTH network.

SN (Sympathizer Node)

A SN is an event receiver like an overhead light, a bell of an alarm clock, on/off of a coffee maker, and so on. Each

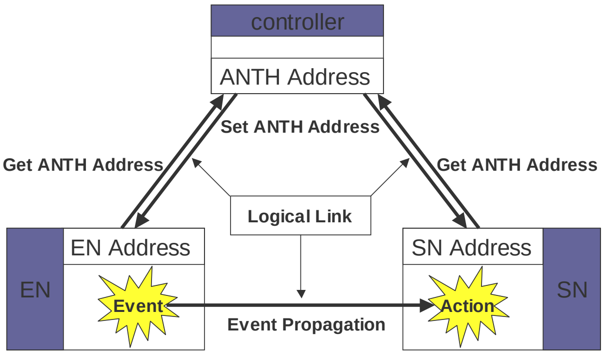

Figure 1: A Bind Control Model

SN holds an ANTH address. When a SN receives an event packet, the SN checks its ANTH address. If the ANTH address of the received packet and the ANTH address of the SN are the same, the SN takes an action.

Controller

A *controller* is a user device that coordinates the relation between an EN and a SN like the mouse on real-space. The *controller* can get and set ANTH addresses of EN and SN freely, and realizes real-space programming by the combination of these operations.

Basic Operations

ANTH application is a set of relations between an EN and a SN, and real-space programming is bundles of the relations. For example, ANTH application is built by describing a relation like "turn on the light if the switch is pushed." This relation between the event like "the switch is pushed" and the action like "turn on the light" is actually the relation between an EN's ANTH address and a SN's ANTH address. Therefore, real-space programming is realized by rewriting ANTH addresses by a *controller*. The *controller's* operations of rewriting ANTH addresses are constructed by combinations of getting or setting EN or SN's ANTH addresses. The combinations can be depicted as following three operations.

Operation 1

Get an EN's ANTH address and set it to a SN.
This operation describes a fundamental relation like "turn on the light if the switch is pushed." This is a standard operation of relation description.

Operation 2

Get a SN's ANTH address and set it to another SN.
Suppose that a user wants to add the relation "ring the bell of the alarm clock if the switch is pushed" after description operation of the above Operation 1. In this case, it is realizable by copying the ANTH address from the SN which can do "turn on a light" to the SN which can do "ring the bell of an alarm clock." This operation is valid when we want to relate two or more SNs with an EN as a group.

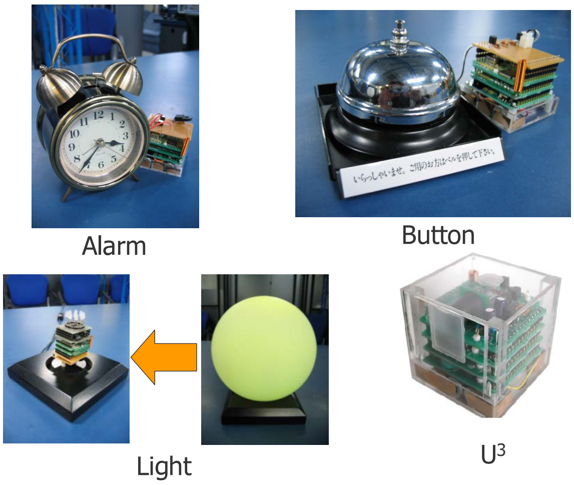

Figure 2: Prototype implementation

Operation 3

Get an EN's ANTH address and set it to another EN.
Suppose that a user wants to change the relation from "turn on the light if the switch is pushed" to "turn on the light if the alarm clock rings" after description operation of the above Operation 1. In this case, it is realizable by copying the ANTH address from the EN which emits the event "the switch is pushed" to the EN which emits the event "the alarm clock timer expires". Moreover, it is possible to erase the original relation by setting a new ANTH address to the EN which has the event "the switch is pushed." This operation is valid when we want to reconstruct the relations efficiently.

As mentioned above, this Bind Control Model enables us to briefly describe user demands like "want to create service by connecting this object and that object" in the abstracted form.

PROTOTYPE IMPLEMENTATION

The ANTH specifies from hardware to user interface. However, the Bind Control Model does not depend on hardware platforms because it is just a part of the ANTH framework and a software mechanism. The Bind Control Model is, additionally, designed to run on tightly limited computing resource. Thus, the Bind Control Model can probably be realized on Smart-Its[1] or MICA[2] hardware platform. Since this prototype implementation is aimed to just check operations of the Bind Control Model and we are very familiar with U^3[3], we have implemented the Bind Control Model on U^3.

Figure 2 shows 4 implemented ANTH objects: a light, an alarm clock, a button, and a U^3which is wireless sensor node. The light and the alarm clock bell work as SNs, the alarm clock timer, the motion sensor of U^3, and the button work as ENs, and laptop acts as a *controller*. We have tested the basic operation of the Bind Control Model using these four types of nodes. For example, the motion sensor event bounded to the alarm clock bell realizes instant security system, which tells us intrusion of someone by ringing the bell.

CONCLUSION AND FUTURE WORK

This paper has described the Bind Control Model and prototype implementation of a real-space programming framework called ANTH (ANtennary THings) for ubiquitous computing environment. Currently, we are developing user-friendly user interface for device cooperation as a *controller*. And, in view of practical use of ANTH, we are also developing communication protocol that concerns low power consumption.

REFERENCES

[1] L. E. Homquist, *et al.*, "Building intelligent environments with smart-its," IEEE Computer Graphics and Applications, vol. 24, no. 1, January-February 2004, pp. 56–64.

[2] J. Hill *et al.*, "Mica: A wireless platform for deeply embedded networks," IEEE Micro, vol. 22, no. 6, November-December 2002, pp. 12–24.

[3] S. SARUWATARI, *et al.*, "Pavenet: A hardware and software framework for wireless sensor networks," Proceedings of the First International Workshop on Networked Sensing Systems (INSS'04), June 2004.